Fermi National Accelerator Laboratory

FERMILAB-Pub-87/190-E
[E-741/CDF]

# FASTBUS Data Acquisition for CDF*

E. Barsotti, A. W. Booth, M. Bowden, J. T. Carroll, C. Day, J. E. Elias,
I. Gaines, L. Giebel, G. Goeransson, H. Gonzalez, B. Graves, M. Haldeman,
D. Hanssen,[1] C. Horvath, M. Larwill, J. Marsh,[2] J. Patrick, D. Quarrie,
C. Rotolo, C. Swoboda, K. Treptow, J. Urish, C. van Ingen,[3] R. Vidal, and S. Zimmerman
*Fermi National Accelerator Laboratory*
Batavia, Illinois 60510 USA

M. Campbell and Y. D. Tsai
*University of Chicago*
Chicago, Illinois 60637 USA

B. Hubbard, F. Kirsten, and J. Siegrist
*Lawrence Berkeley Laboratory*
Berkeley, California 94720 USA

M. Dell'Orso, P. Giannetti, and L. Ristori
*INFN - University of Pisa*
Pisa, Italy 56010

T. Devlin, B. Flaugher, and U. Joshi
*Rutgers University*
New Brunswick, New Jersey 08903 USA

M. Werner
*University of Wisconsin*
Madison, Wisconsin 53706 USA

July-August 1987

[1]Now at Bell Laboratory, Chicago, Illinois
[2]Now at DEC, Chicago, Illinois
[3]Now at SLAC, Stanford, California

*Submitted to Nucl. Instrum. Methods A

# FASTBUS Data Acquisition for CDF†

E. Barsotti, A.W. Booth, M. Bowden, J.T. Carroll, C. Day, J.E. Elias,
I. Gaines, L. Giebel, G. Goeransson, H. Gonzalez, B. Graves, M. Haldeman, D. Hanssen,[1]
C. Horvath, M. Larwill, J. Marsh,[2] J. Patrick, D. Quarrie, C. Rotolo, C. Swoboda,
K. Treptow, J. Urish, C. van Ingen,[3] R. Vidal, S. Zimmerman
*Fermi National Accelerator Laboratory**
Batavia, IL 60510 USA

M. Campbell, Y.D. Tsai
*University of Chicago*
Chicago, IL 60637

B. Hubbard, F. Kirsten, J. Siegrist
*Lawerence Berkeley Laboratory*
Berkeley, CA 94720 USA

M. Dell'Orso, P. Giannetti, L. Ristori
*INFN - University of Pisa*
Pisa, Italy 56010

T. Devlin, B. Flaugher, U. Joshi
*Rutgers University*
New Brunswick, NJ 08903 USA

M. Werner
*University of Wisconsin*
Madison, WI 53706 USA

**Abstract**

All CDF event data is collected in a multilevel FASTBUS network. At the lowest level of this network, MEP/MX and SSP scanners read and buffer data from RABBIT and FASTBUS front end systems. Operation of these front end scanners is coordinated by the Trigger Supervisor module which initiates parallel readout after receiving Level 1 and Level 2 triggers. Dataflow from scanners to consumer processes on host VAX computers is supervised by the Buffer Manager which directs an Event Builder to collect and format data from a set of scanner modules. This system is designed to allow partitioning into semi-independent sections for parallel development and calibration studies.

[1] Now at Bell Laboratory, Chicago, Illinois.
[2] Now at DEC, Chicago, Illinois.
[3] Now at SLAC, Stanford, California.

†Work supported by the United States Department of Energy under contract number DE-AC02-76CH03000 and by the National Science Foundation.
*Operated by Universities Research Association, Inc. under contract with the United States Department of Energy.

# 1. Introduction

The Collider Detector Facility (CDF) at Fermilab is a very large detector for study of $\bar{p}p$ interactions at a center of mass energy of 2 Tev. The detector has 14 major components including central calorimetry, plug and forward calorimetry, vertex tracking, central tracking, central muon and forward muon detectors. These detector components have a total of approximately 75,000 electronic channels. For the run ending in May, 1987 a typical event produced about 80 kilobytes of data with all detector components operational. The $\bar{p}p$ interaction rate is expected to be 50-75 KHz at the design luminosity of $10^{30}$ $cm^{-2}$ $s^{-1}$. CDF has designed a three level trigger system to filter this raw interaction rate down to about 1 Hz which is a practical level for data logging and off-line analysis.

The choice of FASTBUS for the CDF data acquisition system was strongly motivated by its ability to support high data rates into processors which have access to complete event data for on-line triggers. FASTBUS is flexible with support for both high speed devices (synchronous transfers) and low speed devices in asynchronous full handshake transfers. This flexibility and modularity are important in meeting the Data Acquisition (DAQ) requirements of CDF. The FASTBUS protocol with multiple bus masters allows the system to be partitioned so that data acquisition for different detector components can proceed in parallel. Multiple Partitions is a primary concept in the DAQ system for CDF. A partition is one or more sections of the detector that can function essentially independent of other sections. The FASTBUS system for CDF allows dynamic definition of partitions and independent operation from host VAX computers. Use of multiple partitions is particularly important for development tests and detector calibration.

The DAQ and trigger system have a hierarchical structure as shown in Fig. 1. The Level 1 and Level 2 triggers must filter the raw interaction rate down to an acceptance rate of 100 Hz or less -- 100 Hz is a practical limit on the rate at which events can be digitized and buffered. As shown in Fig. 1, most inputs to Level 1 are fast analog signals which are not part of the primary

event data pipeline. The Level 1 decision is made with no deadtime in the 3.5 μs between beam crossings for collider operation with six bunches. At the design luminosity, Level 1 should accept a few KHz of $\bar{p}p$ interactions and Level 2 will reduce this to a rate of about 100 Hz with a deadtime of 5-10%. Both Level 1 and Level 2 use FASTBUS devices which are described elsewhere [1].

When an event is accepted by both Level 1 and Level 2, data from front end crates is digitized and read by scanner modules. CDF has two major types of front end electronics. All calorimetry and central muon detectors use RABBIT front end systems which are read by MX scanners [2]. Most tracking detectors use FASTBUS front end systems which are read by SSP scanners [3]. A few CAMAC modules are included in the event data stream and are read by an SSP scanner using a FASTBUS to CAMAC interface. Each scanner can buffer four events. This step in the DAQ pipeline is managed by the Trigger Supervisor (TS) FASTBUS module. The TS uses a combination of FASTBUS messages and dedicated control lines to provide flexible and efficient control of front end systems. Each partition is allocated a unique set of scanners and a Trigger Supervisor.

When all MX and SSP scanners have finished reading and buffering data for one event, the TS module sends a FASTBUS message to the Buffer Manager indicating that an event is available in a specified buffer. The Buffer Manager (BFM) supervises dataflow from scanner modules to host VAX computers. The BFM initiates this dataflow by sending a FASTBUS message to the Event Builder instructing it to "Pull" an event from the same buffer in all scanners of a specified detector partition. The Event Builder (EVB) is a group of FASTBUS modules which can read, buffer and reformat complete events from any allowed partition of detector components. When the EVB has finished reading (pulling) data from scanner buffer N, it sends a "Pull OK" message to the BFM which in turn notifies the Trigger Supervisor that buffer N is available for a new Level 1/Level 2 trigger.

The next stage in the DAQ pipeline is the Level 3 trigger system which is an array or farm of microprocessors. Level 3 uses VME based processors and control modules developed by the Fermilab Advance Computer Program (ACP) [4]. Under direction of the BFM, the EVB writes a complete event into a specified node in the Level 3 processor farm. For the run ending in May 1987, the Level 3 step in the DAQ pipeline was not implemented and Event Builder functions were performed by a VAX process. Accepted events are read by the Buffer Multiplexor

executing on one or more computers in the VAX cluster. Event data can be logged to disk or tape, and accessed in real time by consumer processes executing on both host VAX computers and remote VAX computers using DECNET. The DAQ system can be operated from any member of the VAX cluster which has a FASTBUS interface.

## 2. FASTBUS Network

The basic elements of the CDF FASTBUS network are crate segments, cable segments and Segment Interconnect (SI) modules which link the cable and crate segments. The first three levels of the CDF network are shown in Fig. 2. This figure shows the primary data paths and module types at the upper network levels -- it is not a complete list of modules in these crate segments. Master modules are shown as boxes and modules which can only function as a FASTBUS slave are indicated by circles.

### 2.1 UPI Module

The UNIBUS Processor Interface (UPI) is the FASTBUS interface to the host VAX computers. As shown at the top of Fig. 2, there are separate UNIBUS interfaces for the B0HOST VAX 785 (UPI H) and the three VAX 750s (UPI A, B and C). The QPI is a Q-Bus version of this interface which is used with the MicroVAX II.

The functional components of the UPI are the FASTBUS Segment Driver (FSD), the FASTBUS Interrupt Receiver (FIR), and the Service Request Receiver (SR). The FSD is a list processing microcoded engine which operates as the FASTBUS master for all I/O operations initiated by the VAX. It is capable of all non-pipeline FASTBUS mastership transactions. The FIR is a FASTBUS slave and generates processor interrupts in response to FASTBUS interrupt messages sent by other FASTBUS masters. The SR generates processor interrupts in response to assertions on the SR line (similar to CAMAC LAMs).

The UPI is composed of two FASTBUS PC boards and one UNIBUS board [5]. The FSD is a 88-bit wide microcoded bit-slice processor which controls two hardware bus master machines, one for UNIBUS DMA transfers and one for FASTBUS segment transactions. The microcode is generally stored in an on-board 4K PROM, but may be downloaded into RAM. A 4K x 32 bit RAM is used for internal parameters and as an intermediate buffer for data transfers

between FASTBUS and UNIBUS. The microcycle is usually set for 528 ns with a main clock of 60 MHz.

A FASTBUS mastership I/O operation is initiated by the host processor writing the address of a control buffer into the I/O registers of the FSD. That control buffer contains the address and length of a command list buffer, a mapped data buffer region, and a status buffer. Each of the three buffers (list, data and status) are limited to 64K bytes, a maximum list length of about 3000 elements. After the control buffer is read, the list execution begins. Each list element corresponds to a full FASTBUS transaction which can be generated by FASTBUS standard routines such as FB_READ_DAT or FB_WRITE_CSR_BLOCK [6]. Each list element writes a status element to the status buffer.

Because of the UPI internal buffering, all FASTBUS block transfers are broken into blocklets of no more than 2K - 1 32-bit words. Blocklets are transferred at full FASTBUS and UNIBUS speeds. The FASTBUS mastership is generally released between blocklets while the data are transferred to the VAX on UNIBUS. FASTBUS DS-DK response times are about 400 ns and UNIBUS transfer times are 4.7 μs per UNIBUS word on the Direct Data Paths. Maximum net throughput from a module in the same crate is approximately 400K bytes/sec.

The UPI was designed as a list processing interface due to the overheads inherent in accessing the device. These overheads include the FORTRAN standard routine interfaces, the VAX QIO, the VAX buffer locking, the UPI initiation, and the UPI list element handling. The net time on a VAX 780 is 3.6 ms for a null command list and 3.9 ms for a single word transaction in delayed execution (pre-built list) mode. The QIO component of these transfer times is 2.5 ms. The VAX buffer locking time is negligible for a single word transfer and depends on the buffer size for a block transfer. The locking time is approximately 32 μs for a 4K buffer and 1.44 ms for a 16K buffer. The UPI microcode initiation overhead is 50 μs; the overhead for each list element is 150 μs. The FASTBUS arbitration, address, and data cycles require only about 2 μs. When the same single word transaction is performed without a pre-built list, the time required on a VAX 11/780 is 6.8 ms -- the difference reflects the management by the Standard Routines. Optimum performance is achieved by pre-building and validating execution lists. The break even point for list construction is two elements; three elements always execute faster if bound to a single list.

## 2.2 Segment Interconnect

The Segment Interconnect (SI) provides a communication path between two otherwise independent, asynchronous FASTBUS segments. It functions as a bidirectional port through which data pass between a backplane crate segment and a cable segment. All standard FASTBUS address and data transactions can be passed through any number of SI modules and segments in a network path. Operations supported by the SI include random transfers to control and data space, broadcast operations, handshake block transfers and pipelined transfers. CDF uses a Model S1 Segment Interconnect which is a multiwire board and complies with all mandatory features of the FASTBUS specification [7,8].

As shown in Fig. 2, SI modules on the same cable segment are daisy-chained together. The cable segment is essentially the same as the crate segment except that the cable segment is implemented using special hybrid differential current drivers. The cable segment bus is provided by two 60-wire twisted-pair flat cables and typical lengths are 20-60 ft for CDF.

Before any communication can take place between segments, the SI's must be initialized. In order for a VAX to access a specific crate or cable segment, all SI modules between the host crate segment and the desired destination must be initialized. Each SI has two route tables which define Group Addresses (GP) which are recognized for passing operations from a near-side to far-side segment. The near-side is the SI port closest to the FASTBUS master which initiates the operation. Each route table has 256 entries corresponding to the 8-bit group field in the upper byte of the 32-bit FASTBUS geographical address. Group addresses for crate and cable segments are shown as two digit hexadecimal numbers in Fig. 2. In order for the UPI in the "HOST CRATE" to access any module in "TS CRATE", the near-side route tables in this path must have pass bits set for entry GP = 02. Bit 1 in CSR 0 of the SI is a global pass bit which must also be set to enable any operation passing.

Times for operation passing through Segment Interconnects are shown in Table 1. Each entry has times to assert and remove the signal. For example, "Crate AS -> Cable AS" is the time between assertion or removal of Address Sync (AS) on the crate segment to assertion or removal of AS on the cable segment. All times in Table 1 were measured with the SI cable length switch set to 50m.

## 2.3 Network Organization

For the run ending in May 1987, CDF used 53 crate segments, 16 cable segments and 66 SI modules. Although FASTBUS modules can be installed on both crate and cable segments, most DAQ modules in the CDF network reside in crates. Each crate can hold a maximum of 26 modules. The CDF system uses over 500 FASTBUS modules of about 35 different types. Most of these module types are PC or multiwire boards which have been custom designed for this experiment.

The majority of segments and modules in the CDF network are used by front end electronics and Level 1/Level 2 trigger systems. Fig. 2 shows network paths from the VAX interface (UPI) to cable segments for each detector component. Detector front end systems are organized in two groups using crate segments FANOUT1 and FANOUT2. Since the hardware Event Builder can read data from these two fan-out crates in parallel, the quantity of data from each fan-out segment should be roughly equal for optimal efficiency. Level 1/Level 2 trigger modules, scalers, latches, trigger summary data and CAMAC modules are accessed via segment AUTOFRED. Although there are 20 crate segments and roughly 170 modules in the trigger system, they produce only 6% of the data for a normal physics trigger. Autonomous FRED is a FASTBUS module which provides an interface to the trigger system for six independent triggers, e.g. calibration and diagnostic triggers. The DAQ system is designed to use two of these modules to support a maximum of 12 "autonomous" partitions. A module called CDF FRED which supports 4 partitions is used for more complicated Level 1/Level 2 physics triggers [1,9].

A network path from the VAX to a front end crate normally has three cable segments and six SI modules. Data from Table 1 show that the round-trip delay for Data Sync to Data Acknowledge is about 700 ns in a path with six SI modules -- this estimate does not include cable segment length and slave response time. Net cable lengths on paths from the host crate to front end crates are 120-180 ft and signal propagation velocity on the cable is 1.6 ns/ft. The DS-DK response time measured at the host crate for a UPI reading a front end scanner module is typically 1.6 μs. The corresponding FASTBUS transit time for a UPI blocklet is about 1.8 μs per 32-bit word. All transactions between the VAX and front end modules use cable segment 81. Most modules in crates connected to cable segment 81 perform DAQ control functions. The Trigger Supervisor, Clock and Crosspoint modules are described in Section 4.

# 3. Front End Systems

CDF requires that all front end modules be read by scanners with generic features as shown in Fig. 3. Higher level masters in the DAQ pipeline access the scanners as logically independent, functionally equivalent FASTBUS slaves. Each scanner has four event buffers which are the first stage in the DAQ pipeline. Each scanner starts execution when a START SCAN word is written to CSR address 8000000E. This START SCAN word includes a 2-bit buffer # and a 6-bit event # which is normally generated by the Trigger Supervisor. START SCAN can be written with broadcast case 2 addressing which defines 16 broadcast classes and allows 16 partitions of the DAQ system to operate simultaneously.

After receiving a START SCAN message, the scanner sets a DONE signal to logic 0 (low). DONE is a differential current mode ECL signal on an external connector. When data collection is complete, DONE is set to logic 1 (high). The event buffer is filled with a 32-bit exclusive word count followed by event data, two pattern words and finally a copy of the START SCAN word. All scanners support pipelined reads of the data buffers. Each event data buffer has a 32-bit buffer status word at a CSR address which is the same for all scanners.

CDF uses MX scanners to access RABBIT front end systems and SSP scanners for FASTBUS based front end electronics. The DAQ design objective called for about 1000 channels per scanner. This will be achieved with approximately 54 MX scanners and 25 SSP scanners.

## 3.1 SSP Scanner

The SSP is a general purpose programmable FASTBUS master and slave on both crate and cable segments [3]. The processor is a microcoded emulator which executes the IBM integer instruction set. The 32-bit CPU is based on AMD 2901C bit-slice technology. All FASTBUS master operations, with the exception of pipelined reads, are implemented by additional microcoded instructions. Separate 4K and 128K x 32 bit memories are provided for program instructions and data respectively; both are accessible via FASTBUS. The SSP has a standard control/status register (CSR 0) and an arbitration level register (CSR 8).

Code may be written in either IBM Assembler H or VS FORTRAN. Object files are produced on an IBM 4381 and sent to the host VAX via a BITNET link. On the VAX, the object files are "translated", that is, the instructions and data are separated, and addresses referenced are

recalculated as a result. Also, a few special cases of non-exact IBM emulation are handled at this stage, and any unsupported instructions flagged. Translated object files are then linked together to produce the executable program which is downloaded to the SSP.

The basic cycle time of the SSP is 120 ns. Most register to register and memory to register instructions require one cycle. Memory stores and conditional branch instructions require two cycles. More complicated instructions require additional cycles. As a FASTBUS master, the time required for the SSP to execute a complete random operation (arbitration, primary address, secondary address and data cycle) is about one microsecond plus the slave response times. When executing block transfers, the delay between the time the SSP receives DK and the time it switches DS is about 135 ns.

The algorithms for each system have a similar general structure. When a start scan broadcast is received, the program begins execution. It reads and processes the data for its assigned devices and raises the DONE signal when complete. If a FASTBUS fault, e.g. bad Slave Status (SS) code, or any other error condition is encountered, the DONE signal is not raised causing a system timeout, and information regarding the error is saved to be read and interpreted by the host VAX.

The algorithm for LeCroy 1879 TDC systems reads data from each 1879 until SS=2 is returned. It reorders the data by channel number and TDC time within each channel. Leading and trailing edge hits are associated, and both times stored in a single word in the output buffer. Pointers to the first word of data for each chamber cell are saved at the beginning of the output block as an aid to off-line reconstruction programs. The time required for this process is about 2 ms of overhead, plus about 9 microseconds/hit. For crates with the largest occupancy (averaging 2000 hits/event), the corresponding scan times average 20 ms. To minimize the system deadtime with higher Level 2 trigger rates, the data reformatting steps can be postponed to higher levels in the DAQ pipeline, reducing the average scan time to around 2 ms.

Other FASTBUS based front end systems read by SSP's include compacted FADC data from the Vertex Time Projection Chamber [10], TDC data from forward muon drift chambers [11] and data from the trigger system. The SSP is also used to read latches, scalers and other miscellaneous data.

### 3.2 MEP/MX Scanner

The MX is a 16-bit integer arithmetic microcomputer which can operate 1-4 RABBIT crates which are located near the detector [2]. During a physics run the MX reads digitized data from the RABBIT system, makes data corrections (threshold suppression, pedestal subtraction and gain corrections) and fills an output buffer in its own memory space. Total event data memory for current MX scanners is 4K x 32 bits. Since RABBIT crates use a 16-bit ADC and the MX is a 16-bit processor, the MX usually formats data as 16-bit words. For production physics runs the MX event memory is used as four 2K x 16 bit buffers. Scanner algorithms are written in MX assembly language and downloaded to the MX using the FASTBUS interface. Algorithm execution times for physics triggers were typically 10-16 ms during the last run.

The Multiple Event Port (MEP) is a single width FASTBUS module with MX control and data interfaces using the FASTBUS auxiliary port connector. There is one MEP for each MX scanner. All MX instruction and data memory can be accessed from the MEP. The MEP supports random, block transfer and pipelined read/writes to MX event data memory. The MEP includes a selective set/clear error register (CSR 1), a broadcast class register (CSR 7), buffer status registers and a START SCAN register. Bits 0-1 in the MEP buffer status register define four buffer states: 0=empty, 1=filling, 2=full and 3=warning. Although the MX normally formats event data in 16-bit words, the MEP always reads/writes MX event buffers in 32-bit words. For example, the MEP FASTBUS secondary address for buffer 0 is 0-3FF which is 1024 32-bit words. The MEP does not have any data memory of its own. For block or pipelined reads, the MEP always pre-fetches the next word from the MX so there is minimal delay for MX access. For a FASTBUS block read, the DS to DK response time measured at the MEP crate segment is 100 ns. A FASTBUS master reading a MX buffer in pipelined mode must allow a minimum interval of 250 ns between DS transitions (interval for 8m of cable between the MEP and MX).

## 4. Trigger Control

### 4.1 Trigger Supervisor

The main function of the Trigger Supervisor [12] is to coordinate the trigger system, the front end scanners (MX and SSP) and the Buffer Manager. A DAQ protocol establishes the sequence of operations leading to the event construction by the Event Builder. It is the job of the Trigger Supervisor (TS) to initiate the elements of the sequence, and monitor the progress. If

any departure from the protocol is detected by the TS, it records in memory the status of each protocol signal for further analysis by higher level software.

The requirements of flexibility and high reliability were met by implementing the TS as TTL sequencers running microcoded programs customized to the DAQ protocol. In the final implementation, the TS consisted of three sequencers and a FASTBUS slave. The C&S sequencer generates the Clear and Strobe gates to the front end electronics, and appropriate queries of the trigger system. The FASTBUS Master sequencer generates the START SCAN FASTBUS Broadcast message to the front end scanners, as well as the FASTBUS BUFFER FULL message to the Buffer Manager, indicating that the scanners have completed digitization and are ready to have one of their buffers read out. The Main Operating Sequencer serves as the overall manager and protocol monitor.

An individual TS is a single FASTBUS card using multiwire board construction, TTL and ECL logic. It has both Master and Slave FASTBUS capabilities and conforms to the full FASTBUS specification. In the current scheme of data acquisition for CDF, the Trigger Supervisors are located in one FASTBUS crate (segment 02 in Fig. 2). There are four Trigger Supervisors for the four possible data-taking partitions, and up to 12 additional Trigger Supervisors for the 12 possible calibration partitions.

### 4.2 Master Clock

The CDF Master Clock provides $\bar{p}p$ bunch crossing related timing signals to the front end electronics [13]. This information is generated in the form of a Clear and Strobe (C&S) pulse with the leading edge (Clear) and the trailing edge (Strobe) both timed relative to the bunch crossing. Absolute timing of C&S relative to the bunch crossing is programmable with 1 ns resolution. The Master clock is synchronized once per revolution to a marker obtained from the Tevatron Beam Sync Clock. Since this marker is present at all times, the Clock can run synchronously with the accelerator independent of actual stored beam.

The Master Clock module is implemented on two FASTBUS boards and occupies 11 slots in a crate segment. The first board, called the Phase Coherent Clock (PCC), produces a 53 MHz internal clock (Pclk) locked in frequency to the Tevatron RF and independently locked in phase once per revolution to a particular proton bunch. This establishes 1113 Pclk cycles per revolution corresponding to the accelerator RF buckets. (For a nominal RF frequency of 53 MHz, one revolution around the ring would take 1113/53 MHz = 21 μs.) The second board

called the Clock Generator is a programmable state generator which synchronously outputs various timing signals at any of the 1113 cycles. These signals are controlled by a program downloaded through FASTBUS.

Distribution of timing signals from the Clock to front end electronics follows two paths to separate timing from control. Continuously generated timing signals including tightly timed versions of C&S are cabled directly from the Clock to several Gate Selector crates located on the major detector components. The control path is through a series of crosspoints and Trigger Supervisors which partition the detector such that a TS has control over the generation of C&S to sections of the detector within a partition. Gate Selectors use a logical AND of C&S signals from these two paths to produce the precisely timed signals actually used for front end gates.

### 4.3 Crosspoint Modules

The DAQ system design allows up to 16 different partitions to run independently and each partition has its own Trigger Supervisor (TS) module. The TS is coupled to a particular trigger and partition of detector components using non-FASTBUS signals which are sent/received through crosspoint modules. As shown in Fig. 4, there are five applications of FASTBUS crosspoint modules. The Master Clock generates three types of C&S gates (beam, calibration and cosmic ray) which are routed to a TS module using a 3x16 START/STOP crosspoint. Registers in this crosspoint are loaded to select the type of gate generated by each TS module. The C&S signals from each TS are input to a 16x80 crosspoint which can be setup to allow any TS module to control gate signals for up to 80 front end systems. The TS-FRED and FRED-TS crosspoints support signal flow between TS and any one of up to 4 Level 1/Level 2 physics triggers (CDF FRED) and 12 independent calibration triggers (AUTO FRED).

The START/STOP, C&S, TS-FRED and FRED-TS crosspoints are all versions of the same FASTBUS module. This standard crosspoint has 16 input channels and 16 output channels. Each channel can contain up to 5 differential ECL signals. Switching is accomplished by connecting output channels to input channels with FASTBUS accessible registers which are loaded using standard support software. Any one of the output channels (with 5 ECL signals) can be connected to any input channel. Input and output cables are attached to the FASTBUS module with front panel connectors. The only difference between these four crosspoint applications is the front panel connector implementation.

The DONE crosspoint is setup from FASTBUS to allow any combination of MX and SSP

scanner done signals to be fanned in to a single output. This makes the ensemble of scanners in a partition appear functionally the same to the TS as a single scanner. The done signal output to TS is a logical OR for "start scan received" and a logical AND for "scan complete". This AND/OR logic feature requires a different type of crosspoint than the above modules. The DONE crosspoint consists of three boards: a FASTBUS slave interface/control board and two identical scanner input boards. Each scanner input board has 64 scanner done inputs and the FASTBUS interface/control board has 16 scanner done outputs. The DONE module includes diagnostic registers to check the AND and OR case for each partition, disable/enable done outputs and examine the latched done status for scanners in each partition.

# 5. Event Readout

## 5.1 Buffer Manager

The Buffer Manager is the master intelligence controlling the flow of data through the DAQ pipeline. It has the responsibility for scheduling the processing elements at each stage (known as Pipeline Stage Elements or PSEs) and maintaining tables of statistics for each. In order to provide flexibility in configuring the system, PSEs may be dynamically added to or removed from the pipeline such that the availability of another Event Builder, for example, may be quickly and painlessly utilized. A further requirement is that the dataflow should be independent of whether part or all of the pipeline is implemented in FASTBUS hardware or in software on a VAX. During the run ending in May 1987, for example, the hardware Event Builder was not yet operational and its job was performed by a process running in the VAX. Likewise, there was no VAX process substituting for the prototype Level 3 system and the Level 3 PSE was omitted from the pipeline. The functionality of the various PSEs and the protocols by which they communicate with the Buffer Manager has been decoupled as much as possible from their implementation and the transmission medium via which they communicate. Similarly the Buffer Manager itself has been initially implemented as a process on the host VAX.

The Buffer Manager commands the PSEs to move the data across FASTBUS; it does not move the data itself. For PSEs actually implemented on FASTBUS, each command is contained within a single 16-longword FASTBUS interrupt message. The Buffer Manager also is the device which decides which events are actually recorded. Messages from PSEs contain trigger mask bits which classify the event on criteria analyzed either by the trigger system or by Level 3. The Buffer Manager maintains a list of required trigger bits for each Consumer Computer in the pipeline. The Buffer Manager will command each Consumer Computer with a requirement satisfied by a particular event to read that event. When all consumers have finished reading, the event is released. If no consumer wants the event, the Buffer Manager releases the event immediately. Since the Buffer Manager is the only device which knows the fate of every event in the pipeline, it is the only one which can keep accurate statistics of acceptance rates.

Fig. 5 shows all of the dataflow messages sent and received by the Buffer Manager to move a single event through a pipeline. The pipeline in this case has four stages, Trigger Supervisor, Event Builder, Level 3 and Consumer Computer, and the event is assumed to pass one of the trigger mask requirements. Time flow is from top to bottom of the figure. These are the

messages required to move only one event. The pipeline will actually have multiple events in it at various PSEs. The message sequence for any given event may look like Fig. 5, but the sequence as seen by the Buffer Manager will be an interlaced set from all of the events.

At the design rate, the messages from the Trigger Supervisor NEW EVENT through the Level 3 PROCESSING DONE will occur at a rate of 100 Hz each. In order to handle this rate, the current Buffer Manager software running on the host VAX will be moved to a dedicated MicroVAX II under VAXEln. This will remove the operating system overheads of VMS; the code itself should already be fast enough now to handle the rate. Finally, the microcode of the QPI interface for the Buffer Manager and the corresponding software driver will be modified to communicate through a shared memory ring buffer. This will eliminate the overhead of an interrupt context switch except in the case of an empty or full ring buffer. In principle, even higher rates could be handled by stacking two Buffer Managers, one to move the events from the front end into the Event Builder and another to move the events out of the Event Builder, through Level 3 and into the Consumer Computers.

### 5.2 Event Builder

The hardware Event Builder [14] is designed to collect data from front end scanners and write complete event records to a processor in the Level 3 system with a throughput of 100 events/sec. The Event Builder (EVB) reads calibration or event data from any partition of MX and SSP scanners. EVB operations are executed in response to a FASTBUS message received from the Buffer Manager. After receiving a PULL EVENT message, the EVB reads the designated buffer from all scanners in the partition specified by the Buffer Manager. The EVB can reformat scanner data into banks which are organized by detector component and have the same format expected by VAX tape logger and consumer processes. When it receives a PUSH EVENT message, the EVB writes reformatted data for a complete event to a Level 3 processor designated in the Buffer Manager control message. The reformat processing step only produces bank headers in the YBOS format used by CDF and tables of internal EVB pointers to data for each detector component. (YBOS is a memory management system with a data structure organized in subsections called banks.) As it writes to Level 3, the EVB uses these internal tables to gather data for each detector component into contiguous banks.

The initial hardware EVB configuration has a central controller on the FASTBUS crate

segment and an auxiliary controller on each of the two fan-out cable segments 82 and 84 shown in Fig. 6. (The EVB connection to cable segments 82 and 84 is not shown in Fig. 2.) Each cable segment controller has one or more reformatter boards which receive data from the cable segment, reformat event data and write to Level 3 on the EVB crate segment shown in Fig. 6. For I/O operations on the cable (crate) segment, the reformatter board operates under control of the EVB cable (crate) segment controller. The reformatter board is double buffered and can perform any two of its read, reformat and write functions in parallel. The central control board handles all communication with the Buffer Manager and supervises the cable controller and reformatter boards using a front panel control bus. All three EVB board types use Motorola MC68020 microprocessors.

The initial EVB configuration with five boards (one reformatter on each fan-out cable segment) should support rates of 30-50 Hz into Level 3 for a nominal event size of 100K bytes. To provide these data rates, the EVB must read scanner modules and write to Level 3 with FASTBUS pipelined transfers. Higher throughput can be achieved with additional reformatter boards. A configuration with three reformatter boards on each FASTBUS fan-out cable should support rates of 90-150 Hz.

### 5.3 Level 3 System

The Level 3 System is designed to execute FORTRAN-77 filter algorithms as the last stage in on-line trigger selection. Level 3 uses ACP 32-bit processors installed in VME crates with VME bus control and interface modules [4]. The processors are single width double height VME cards based on the Motorola 68020 CPU and 68881 floating point coprocessor. Benchmark studies show the processing capacity of an ACP processor to be about 67% of a VAX 780. For the next run each processor will have 4M bytes of DRAM -- processors in the prototype system have 2M bytes.

Multiple VME crates are daisy-chained to I/O controllers using the ACP Branch Bus. The Branch Bus is a 32-bit data bus with a non-handshake pipelined protocol. It is implemented as two 50-wire twisted pair flat cables. As shown in Fig. 6, the Event Builder and host VAX use the FASTBUS to Branch Bus Controller (FBBC) for I/O to Level 3 [15]. The FBBC is a FASTBUS slave which provides transparent management of differences in bus protocol. A target VME crate can be selected by a control byte in CSR 10 of the FBBC or with segment

extended addressing. The FBBC data space secondary address is output as the VME address, e.g. node number and memory address. The FBBC has logic for byte and 16-bit halfword swapping in the 32-bit data path to allow transformation between DEC and IBM/Motorola byte ordering on each read or write transfer. The FBBC and Branch Bus support a transfer rate of 20M bytes/sec into Level 3 processor memory. I/O to the prototype Level 3 system has been throughly tested using diagnostics which execute on both the VAX and on the SSP FASTBUS module.

The Level 3 processing farm is managed by a process called the Farm Steward which executes on a dedicated MicroVAX II. This MicroVAX II has a QPI interface to FASTBUS and a Q-Bus Branch Bus interface to multiple VME crates. The Farm Steward can download Level 3 processors with a filter algorithm and calibration data when it receives a DECNET control message to initialize the processors. While a run is in progress, the Farm Steward communicates with the Buffer Manager via FASTBUS messages. The Farm Steward reads a register in each VME crate to check the status of all nodes which have received an event from the Event Builder and started filter algorithm execution. When a node completes execution, the Farm Steward reads a 128-bit trigger mask from processor memory and sends this trigger summary to the Buffer Manager in CDF standard 16 word message format. Level 3 is designed to support accepted event rates of a few Hz.

## 6. DAQ Operation

To start operation of the DAQ system the user must define the partition, i.e. the detector components and corresponding MX and SSP scanners which are to be included. A software resource management system prevents the same component from being allocated to more than one partition. For a normal physics run, the Level 1/Level 2 trigger is defined by selecting a physics table which specifies the filters to be used and their associated parameters, e.g. cuts on energy deposited in the calorimeter, requirement of a beam-beam counter coincidence, rate limits, etc. A look-up table is generated from this card image physics table and downloaded to CDF FRED. Additional initialization is required for the central and forward muon triggers.

For MX scanners included in the partition, readout lists are generated from a RABBIT database and tables of pedestal, gain and threshold parameters are produced from calibration and

configuration databases. Each MX is downloaded from its MEP interface to FASTBUS and the programs are read and verified. Information for SSP readout lists is obtained from a database and the FASTBUS resource management system. An algorithm plus readout list is loaded and verified for each SSP in the partition. The Event Builder is loaded with a list of MX and SSP scanners for readout.

The Trigger Supervisor is loaded with a unique broadcast address (class) for the partition, the secondary address for a START SCAN broadcast and the Buffer Manager primary and secondary FASTBUS addresses. CSR 7 in MX scanners and CSR 0 in SSP scanners are loaded with the same broadcast class. Crosspoint modules are setup to connect the clock, front end systems, Level 1/Level 2 and scanner DONE signals with the Trigger Supervisor allocated to this partition. When the scanners, Trigger Supervisor and crosspoint modules have been setup, the VAX run control process issues an initialization START SCAN to check that all scanners for this partition receive the FASTBUS broadcast and the DONE signal is properly returned. In response to this initialization START SCAN, the MX and SSP scanners execute setup procedures for their front end systems in parallel.

All of these DAQ initialization tasks are performed under control of processes executing on a host VAX. When Level 3 is included in the DAQ pipeline, the Farm Steward process will be started on its MicroVAX II and initialize Level 3 processors in parallel with the above operations. Most of these initialization steps are unnecessary if the hardware configuration and trigger are unchanged from the previous run.

During run initialization the Trigger Supervisor Main Operating Sequencer is paused. Data acquisition is enabled with a write to Trigger Supervisor CSR 0 which puts it in an active/continue state. The Trigger Supervisor generates C&S gates to its front end systems until a Level 1 Accept (L1A) is received. Upon receiving L1A the Trigger Supervisor stops generating gates and waits for a Level 2 Accept (L2A). When both L1A and L2A are present, it broadcasts a START SCAN message, waits for DONE from all scanners in the partition and finally sends a NEW EVENT message to the Buffer Manager. The Trigger Supervisor can then continue to generate front end gates and accept triggers unless all four scanner buffers are full.

The Buffer Manager controls the DAQ pipeline from scanners to consumer VAX with a

sequence of FASTBUS messages (see Section 5.1). As shown in Fig. 5, the Buffer Manager can tell the Trigger Supervisor that a partition scanner buffer is free as soon as it receives a PULL OK message from the Event Builder. However, the Buffer Manager must wait for a PROCESSING DONE message from the Event Builder, i.e. reformatting complete, before sending a control message to push the event into a Level 3 processor. Since this push message specifies the Level 3 node to be used, both the Buffer Manager and the Farm Steward must have access to the list of operational Level 3 processors.

A run in progress is paused by setting a control bit in Trigger Supervisor CSR 0. A run can be terminated by disabling use of additional scanner buffers and allowing the normal Buffer Manager control sequence to clear the DAQ pipeline of any triggers remaining in scanner buffers, Event Builder buffers or Level 3 processors. The scanners and Trigger Supervisor do not have any end run functions. Both the Event Builder and Level 3 could potentially provide performance summary information for VAX consumer processes at the end of a run.

## 7. Summary

For the run ending in May, 1987 the CDF DAQ system was operational with 1-3 partitions using the VAX 785 and two of the VAX 750s. The third VAX 750 (B0SCCC) was used for the CDF Alarms and Limits system. The Buffer Manager and Event Builder were implemented as VAX processes with control and readout procedures compatible with plans for hardware versions of these tasks. Since a VAX process accesses FASTBUS via the UPI interface, the network was not tested with pipelined transfers through multiple segments. Typical event rates were about 0.6 Hz. A prototype Level 3 system was operational and could be tested as a consumer of accepted events while a run was in progress -- it was not integrated into the DAQ pipeline.

For the next run, the number of MX and SSP scanners will be increased to match the original CDF design plan and scanner algorithm efficiency will be improved to support higher rates into Level 3. The Buffer Manager will execute on a dedicated MicroVAX II and the hardware Event Builder will be operational with one reformatter board on each fan-out cable segment. This configuration should support event rates of about 30 Hz into a Level 3 system with 50 processors fully integrated into the DAQ pipeline.

## References


1. D. Amidei et al., "A Two Level Fastbus-Based Trigger System for CDF", submitted to Nuclear Instruments and Methods.

2. G. Drake et al., "Front End Electronics: The RABBIT System," submitted to Nuclear Instruments and Methods.

3. H. Brafman et al., "The SLAC Scanner Processor: A FASTBUS Module For Data Collection And Processing", IEEE Trans. Nucl. Sci. NS-32 (1985) 336.

4. H. Areti et al., "The ACP Multiprocessor System at Fermilab", Proceedings of the XXIII International Conference on High Energy Physics, Berkeley, California (1986).

5. M. Larwill et al., "Data Collection from FASTBUS to a DEC UNIBUS Processor Through the UNIBUS-Processor Interface", IEEE Trans. Nucl. Sci. NS-30 (1983) 4003.

6. S. Gannon et al., "Proposed Revision of the FASTBUS Standard Routines", Proceedings of the FASTBUS Software Workshop, CERN 85-15 (1985) 108.
   R. Pordes, "Review of the Status of the FASTBUS Standard Routine Specification", IEEE Trans. Nucl. Sci. NS-34 (1986) 162.

7. "IEEE Standard FASTBUS Modular High-Speed Data Acquisition and Control System", The Institute of Electrical and Electronics Engineers, Inc., New York, 1985.

8. C. Swoboda et al., "Status of the Segment Interconnect, Cable Segment Ancillary Logic, and the Cable Segment Hybrid Driver Projects", IEEE Trans. Nucl. Sci. NS-32 (1985) 1335.

9. M. Campbell et al., "CDF Trigger Interface Board FRED", IEEE Trans. Nucl. Sci. NS-32 (1985) 1345.

10. F. Snyder et al., "The CDF Vertex Time Projection Chamber System", submitted to Nuclear Instruments and Methods.

11. D. Carlsmith et al., "The CDF Forward Muon System", submitted to Nuclear Instruments and Methods.

12. E. Barsotti et al., "The Trigger Supervisor", IEEE Trans. Nucl. Sci. NS-32 (1985) 1348.

13. C.J. Rotolo et al., "The Collider Detector Master Clock System", CDF Note 514 (1986).

14. A.W. Booth et al., "The CDF Event Builder", Proceedings of The Fifth Conference on Real-Time Computer Applications in Nuclear, Particle and Plasma Physics, San Francisco (1987).

15. J.T. Carroll et al., "Level 3 System at CDF", Proceedings of the 3rd Pisa Meeting on Advanced Detectors, Castiglione della Pescaia, Italy (1986).
B. Flaugher et al., "Integration of the ACP Multiprocessor Farm with the CDF FASTBUS Data Acquisition System", Proceedings of The Fifth Conference on Real-Time Computer Applications in Nuclear, Particle and Plasma Physics, San Francisco (1987).

## Figure Captions

Fig. 1. DAQ Pipeline.

Fig. 2. FASTBUS Network.

Fig. 3. Scanner FASTBUS Interface.

Fig. 4. Crosspoint Modules.

Fig. 5. Buffer Manager/Pipeline Message Traffic.

Fig. 6. DAQ Upgrade with Buffer Manager, Event Builder and Level 3.

## Table 1: SI Operation Passing Times

| OPERATION | ASSERT TIME | REMOVAL TIME |
|---|---|---|
| Crate AS → Cable AS | 170 ns | 86 ns |
| Crate AK → Cable AK | 92 ns | 90 ns |
| Crate DS → Cable DS | 94 ns | 84 ns |
| Crate DK → Cable DK | 80 ns | 82 ns |
| Crate AD → Cable AD | 19 ns | 17 ns |
| | | |
| Cable AS → Crate AS | 88 ns | 26 ns |
| Cable AK → Crate AK | 24 ns | 28 ns |
| Cable DS → Crate DS | 24 ns | 20 ns |
| Cable DK → Crate DK | 36 ns | 38 ns |
| Cable AD → Crate AD | 11 ns | 11 ns |

AS, AK = Address Sync, Address Acknowledge
DS, DK = Data Sync, Data Acknowledge
AD = Address/Data lines

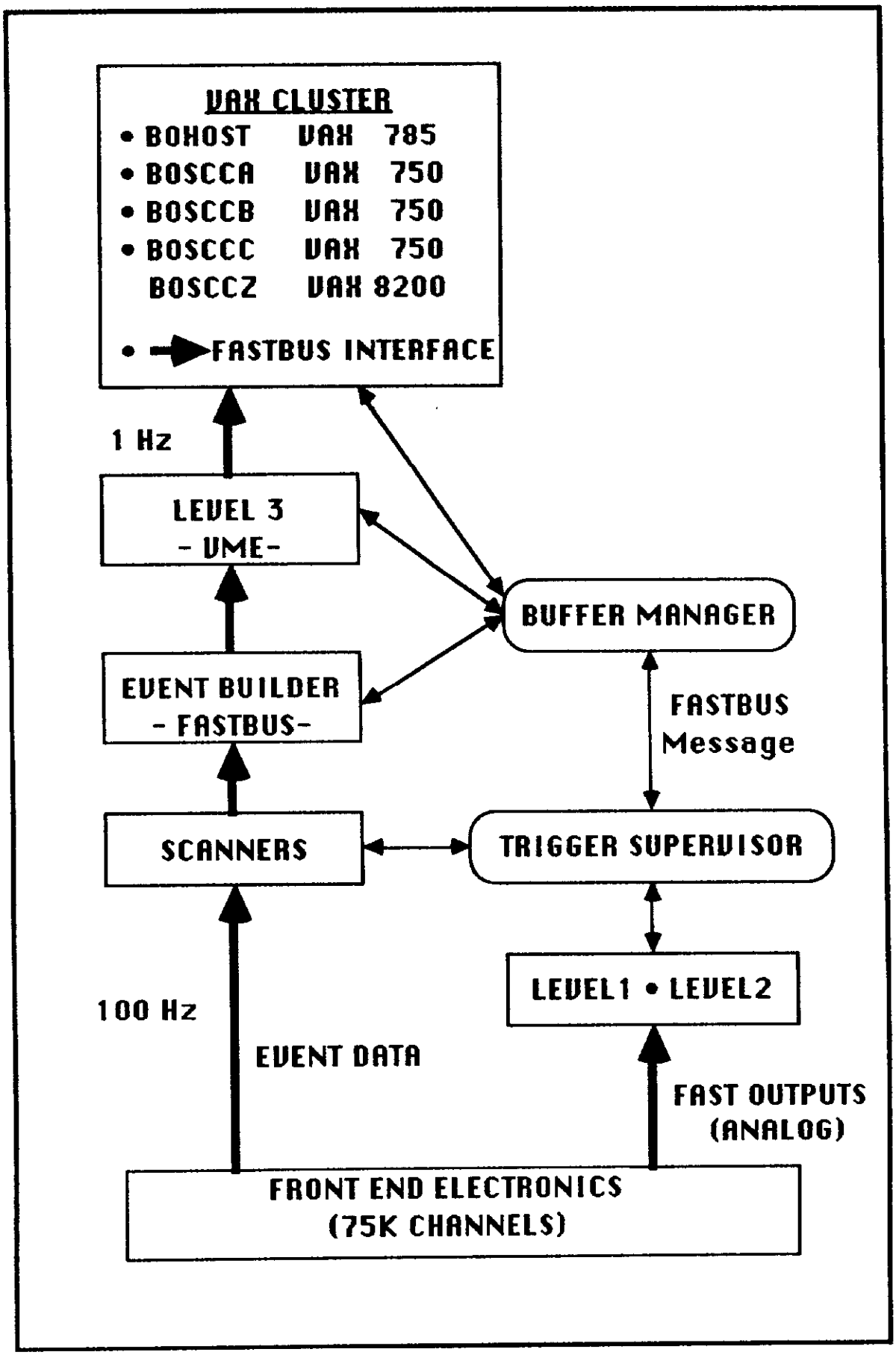
VAX CLUSTER
• BOHOST VAX 785
• BOSCCA VAX 750
• BOSCCB VAX 750
• BOSCCC VAX 750
BOSCCZ VAX 8200
• FASTBUS INTERFACE
1 Hz
LEVEL 3
- UME-
BUFFER MANAGER
EVENT BUILDER
- FASTBUS-
FASTBUS
Message
SCANNERS
TRIGGER SUPERVISOR
LEVEL1 • LEVEL2
100 Hz
EVENT DATA
FAST OUTPUTS
(ANALOG)
FRONT END ELECTRONICS
(75K CHANNELS)

Fig 1

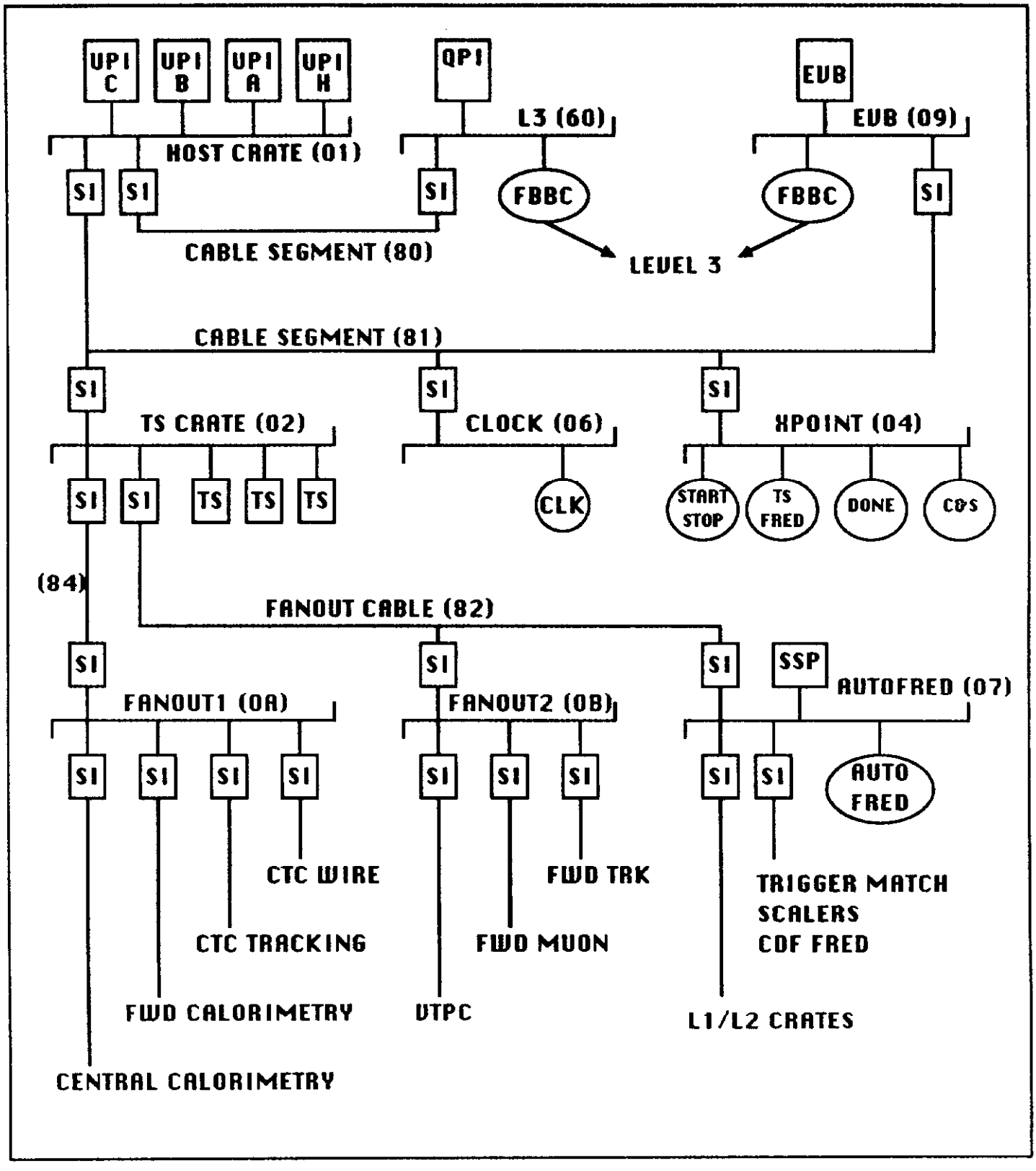
UPI C
UPI B
UPI A
UPI H
QPI
EVB
L3 (60)
EVB (09)
HOST CRATE (01)
SI
FBBC
CABLE SEGMENT (80)
LEVEL 3
CABLE SEGMENT (81)
TS CRATE (02)
CLOCK (06)
XPOINT (04)
TS
CLK
START STOP
TS FRED
DONE
C&S
(84)
FANOUT CABLE (82)
FANOUT1 (0A)
FANOUT2 (0B)
SSP
AUTOFRED (07)
AUTO FRED
CTC WIRE
CTC TRACKING
FWD CALORIMETRY
CENTRAL CALORIMETRY
FWD TRK
FWD MUON
VTPC
TRIGGER MATCH
SCALERS
CDF FRED
L1/L2 CRATES

Fig 2

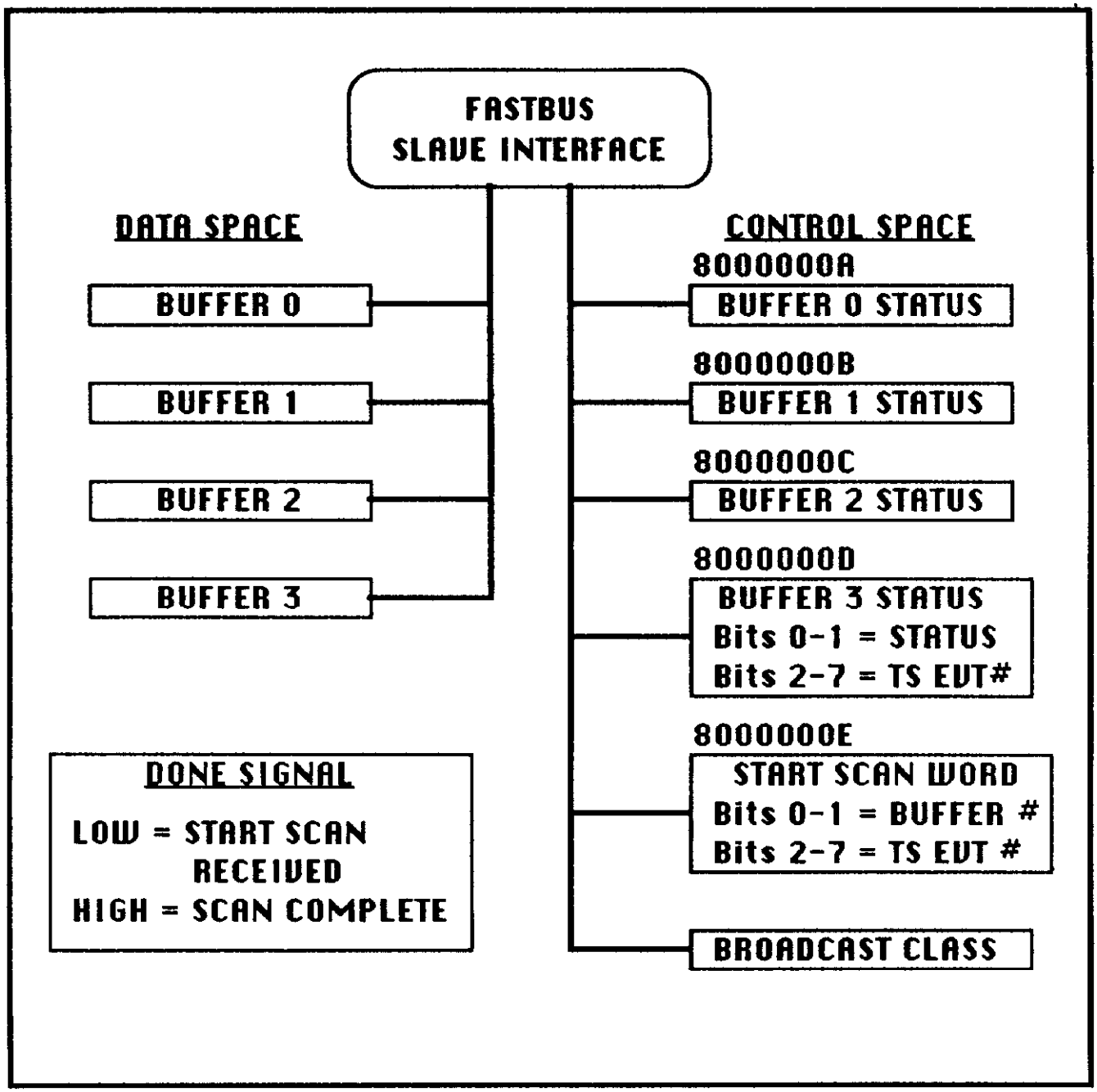
FASTBUS
SLAVE INTERFACE
DATA SPACE
BUFFER 0
BUFFER 1
BUFFER 2
BUFFER 3
CONTROL SPACE
8000000A
BUFFER 0 STATUS
8000000B
BUFFER 1 STATUS
8000000C
BUFFER 2 STATUS
8000000D
BUFFER 3 STATUS
Bits 0-1 = STATUS
Bits 2-7 = TS EVT#
8000000E
START SCAN WORD
Bits 0-1 = BUFFER #
Bits 2-7 = TS EVT #
BROADCAST CLASS
DONE SIGNAL
LOW = START SCAN RECEIVED
HIGH = SCAN COMPLETE

Fig 3

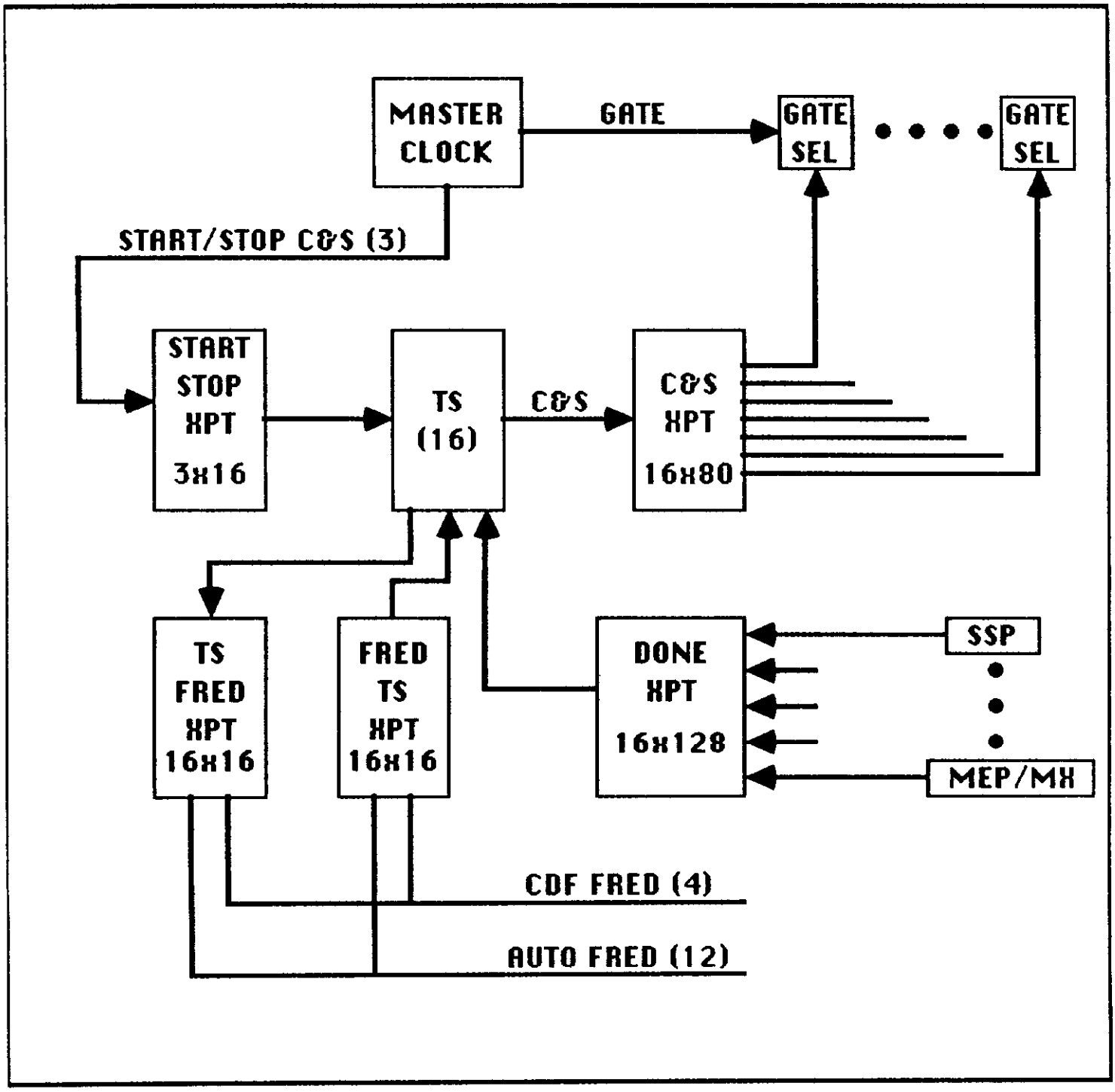
MASTER CLOCK
GATE
GATE SEL
GATE SEL
START/STOP C&S (3)
START STOP XPT 3x16
TS (16)
C&S
C&S XPT 16x80
TS FRED XPT 16x16
FRED TS XPT 16x16
DONE XPT 16x128
SSP
MEP/MX
CDF FRED (4)
AUTO FRED (12)

Fig 4

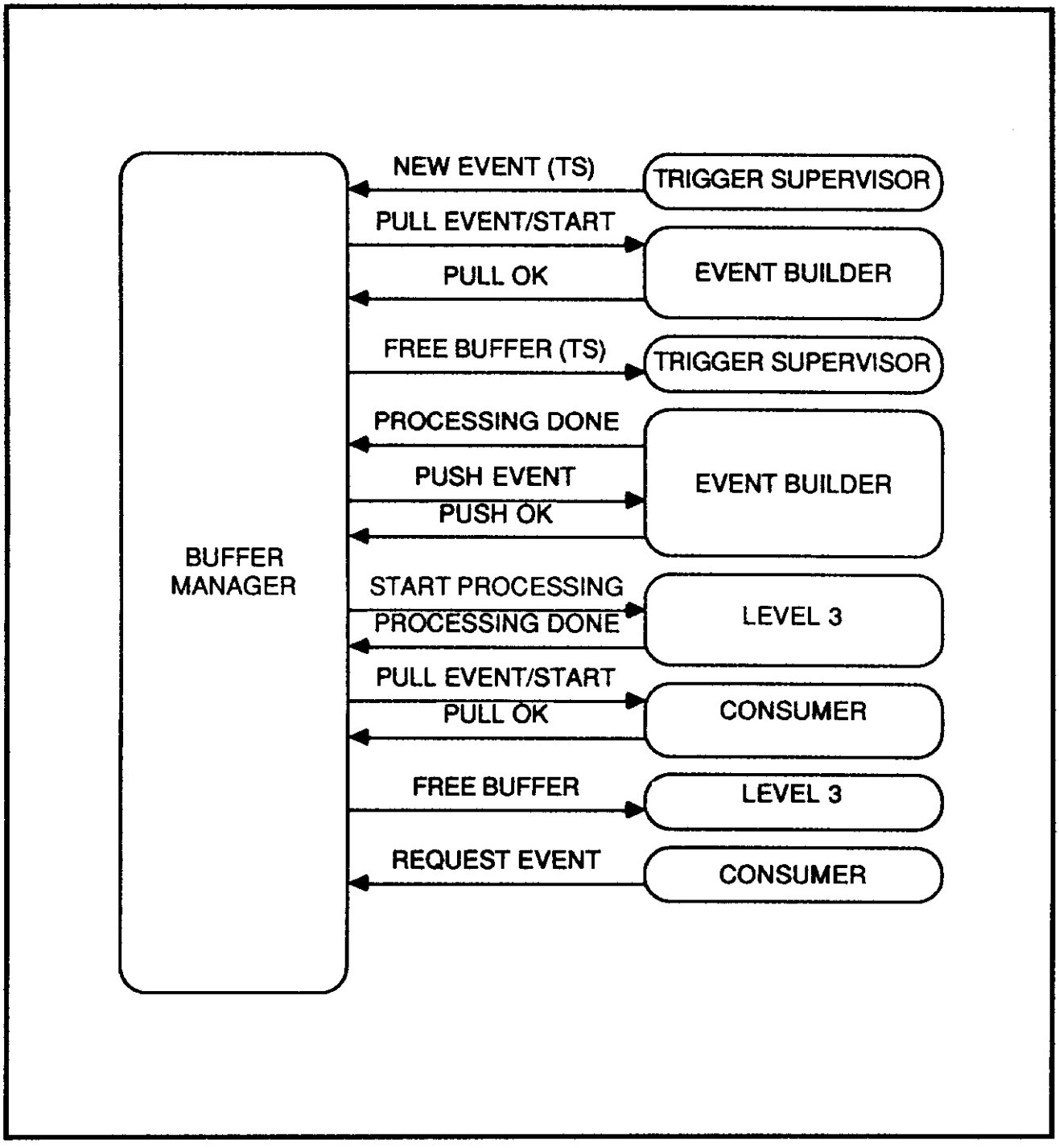
BUFFER MANAGER
NEW EVENT (TS)
TRIGGER SUPERVISOR
PULL EVENT/START
PULL OK
EVENT BUILDER
FREE BUFFER (TS)
TRIGGER SUPERVISOR
PROCESSING DONE
PUSH EVENT
PUSH OK
EVENT BUILDER
START PROCESSING
PROCESSING DONE
LEVEL 3
PULL EVENT/START
PULL OK
CONSUMER
FREE BUFFER
LEVEL 3
REQUEST EVENT
CONSUMER

Fig 5

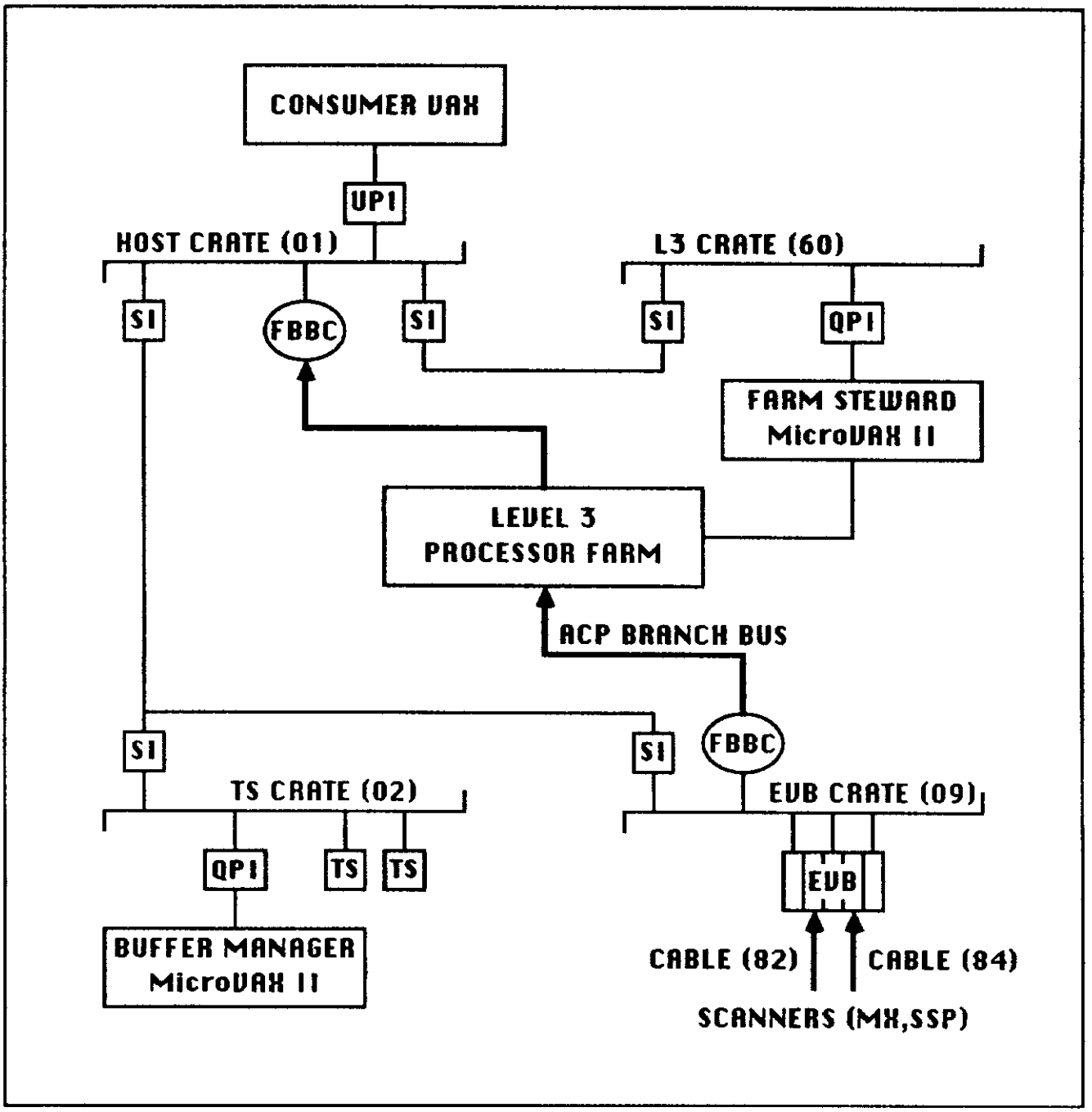
CONSUMER VAX
UP1
HOST CRATE (01)
L3 CRATE (60)
SI
FBBC
SI
SI
QP1
FARM STEWARD
MicroVAX II
LEVEL 3
PROCESSOR FARM
ACP BRANCH BUS
SI
SI
FBBC
TS CRATE (02)
EVB CRATE (09)
QP1
TS
TS
EVB
BUFFER MANAGER
MicroVAX II
CABLE (82)
CABLE (84)
SCANNERS (MX,SSP)

Fig 6